ONLINE BLOOD DONATION MANAGEMENT SYSTEM PROJECT REPORT.

1 author:

Kamal Acharya
Tribhuvan University
192 PUBLICATIONS   **2,699** CITATIONS

AN

INTERNSHIP REPORT

ON

ONLINE BLOOD DONATION MANAGEMENT SYSTEM PROJECT

BY

KAMAL ACHARYA

(Tribhuvan University)

Date: 2023/07/10

TABLE OF CONTENTS

Page

DECLARATION

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Signature
Name
Roll No.
Date.

ABSTRACT

Blood Donation Management System is a web database application that enables the public to make online session reservation, to view nationwide blood donation events online and at the same time provides centralized donor and blood stock database. This application is developed by using ASP.NET technology from Visual Studio with the MySQL 5.0 as the database management system. The methodology used to develop this system as a whole is Object Oriented Analysis and Design; whilst, the database for BDMS is developed by following the steps in Database Life Cycle. The targeted users for this application are the public who is eligible to donate blood ,'system moderator, administrator from National Blood Center and the staffs who are working in the blood banks of the participating hospitals. The main objective of the development of this application is to overcome the problems that exist in the current system, which are the lack of facilities for online session reservation and online advertising on the nationwide blood donation events, and also decentralized donor and blood stock database. Besides, extra features in the system such as security protection by using password, generating reports, reminders of blood stock shortage and workflow tracking can even enhance the efficiency of the management in the blood banks. The final result of this project is the development of web database application, which is the BDMS.

INTRODUCTION

1.1. Project Overview

The Blood Donation Agent is to create an e-Information about the donor and organization that are related to donating the blood. Through this application any person who is interested in donating the blood can register himself in the same way if any organization wants to register itself with this site that can also register. Moreover if any general consumer wants to make request blood online he can also take the help of this site. Admin is the main authority who can do addition, deletion, and modification if required.

1.2. Project Description

This project is aimed to developing a voluntary Blood Donation Information. The entire project has been developed keeping in view of the distributed client server computing technology, in mind.

The Blood Donation Agent is to create an e-Information about the donor and organization that are related to donating the blood. Through this application any person who is interested in donating the blood can register himself in the same way if any organization wants to register itself with this site that can also register. Moreover if any general consumer wants to make request blood online he can also take the help of this site.

Admin is the main authority who can do addition, deletion, and modification if required.

The project has been planned to be having the view of distributed architecture, with centralized storage of the database. The application for the storage of the data has been planned. Using the constructs of MS-SQL Server and all the user interfaces have been designed using the ASP.Net technologies.

The database connectivity is planned using the "SQL Connection" methodology. The standards of security and data protective mechanism have been given a big choice for proper usage.

The application takes care of different modules and their associated reports, which are produced as per the applicable strategies and standards that are put forwarded by the administrative staff.

The entire project has been developed keeping in view of the distributed client server computing technology, in mind. The specification has been normalized up to 3NF to eliminate all the anomalies that may arise due to the database transaction that are executed by the general users and the organizational administration. The user interfaces are browser specific to give distributed accessibility for the overall system. The internal database has been selected as MS-SQL server 2000.

The basic constructs of table spaces, clusters and indexes have been exploited to provide higher consistency and reliability for the data storage. The MS-SQL server 2000 was a choice as it provides the constructs of high-level reliability and security. The total front end was dominated using the ASP.Net technologies. At all proper levels high care was taken to check that the system manages the data consistency with proper business rules or validations.

The database connectivity was planned using the latest "SQL Connection" technology provided by Microsoft Corporation. The authentication and authorization was crosschecked at all the relevant stages. The user level accessibility has been restricted into two zones namely.

Problem Definition

2.1 Existing System

- Cannot Upload and Download the latest updates.
- No use of Web Services and Remoting.
- Risk of mismanagement and of data when the project is under development.
- Less Security.
- No proper coordination between different Applications and Users.
- Fewer Users – Friendly

Disadvantages

1. User friendliness is provided in the application with various controls.
2. The system makes the overall project management much easier and flexible.
3. Readily upload the latest updates, allows user to download the alerts by clicking the URL.
4. There is no risk of data mismanagement at any level while the project development is under process.
5. It provides high level of security with different level of authentication.

2.2. Proposed System

To debug the existing system, remove procedures those cause data redundancy, make navigational sequence proper. To provide information about audits on different level and also to reflect the current work status depending on organization/auditor or date. To build strong password mechanism.

Advantages:

- User friendliness I provided in the application with various controls.
- The system makes the overall project management much easier and flexible.
- Readily upload the latest updates ,allows user to download the alerts by clicking the url. It provides high level of security with different level of   authentication.

Feasibility Study

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility
- Economical Feasibility

3.1. Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users.

The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

The software and hard requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source. The work for the project is done with the current equipment and existing software technology. Necessary bandwidth exists for providing a fast feedback to the users irrespective of the number of users using the system.

3.2. Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

3.3. Economical Feasibility

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

System Analysis

4.1.   Software Requirement Specification (SRS)

The software, Site Explorer is designed for management of web sites from a remote location.

INTRODUCTION

Purpose: The main purpose for preparing this document is to give a general insight into the analysis and requirements of the existing system or situation and for determining the operating characteristics of the system.

Scope: This Document plays a vital role in the development life cycle (SDLC) and it describes the complete requirement of the system. It is meant for use by the developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

DEVELOPERS   RESPONSIBILITIES   OVERVIEW:

The developer is responsible for:

- Developing the system, which meets the SRS and solving all the requirements of the system?
- Demonstrating the system and installing the system at client's location after the acceptance testing is successful.
- Submitting the required user manual describing the system interfaces to work on it and also the documents of the system.
- Conducting any user training that might be needed for using the system.
- Maintaining the system for a period of one year after installation.

The modules involved are:

1. **Administration:**

   In this module the Administrator has the privileges to add all the Blood Groups, Type, State, District, and Location. He can search all the info about the Organization, Donor.

User Account:

- Name
- Username
- Password

Functionality

- Association User Account with UserRole.
- Association User Account with personal Details.
- Association User Account with Blood Donation Details.

Alerts:

- All fields are mandatory
- Select unique username
- Select unique password

Alerts:

- Select Role Id
- Select role name

BDA State:

- State ID
- State Name

Functionality:

- Association state with city
- Association state with Address

Alerts:

- Select state Id
- Select state name

BDA City:

- City ID
- City Name

Alerts:

- Select city Id
- Select city Name

Blood Group:

- Blood Group ID
- Blood Group
- Description
- Active

Functionality:

- Association Blood group with Personal details.

Alerts:

- Select Blood Group ID

Blood Type:

- Blood Type
- Type Name

Functionality:

- Association Blood type with Personal details.

Alerts:

- Select Blood Group
- Select Type Name

Personal Details:

- ID
- Name
- Email
- DOB
- Gender
- Blood Type
- Address
- Contact No

Functionality:

- Association personal detaials with preferd location.

Alerts:

- Select user account
- Select Email id
- Select date of birth

Donor:

Donor is that person who is interested in donating their blood so they can register themselves through this website. If any requirement comes then they will be contacted and they can donate their blood. Along with it they can search for the various organization locations wise and can also make request for blood if needed

Donor:

- User Account

Functionality:

- Association Donor preferred organization with personal details.

Alerts:

- Select user account

4.2. HARDWARE REQUIREMENTS:

- PIV 2.8 GHz Processor and Above
- RAM 512MB and Above
- HDD 20 GB Hard Disk Space and Above

4.3. SOFTWARE REQUIREMENTS:

- WINDOWS OS (XP / 2000 / 200 Server / 2003 Server)
- Visual Studio .Net 2005 Enterprise Edition
- Internet Information Server 5.0 (IIS)
- Visual Studio .Net Framework (Minimal for Deployment)
- SQL Server 2000 Enterprise Edition

System Design

5.1. Data Flow Diagrams (DFD)

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams.

The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose.

The development of DFD'S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design.

A DFD is also known as a "bubble Chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

DFD SYMBOLS:

In the DFD, there are four symbols

1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data
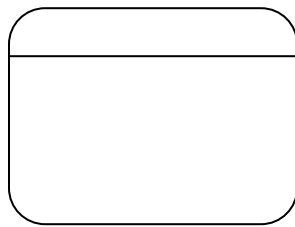
Process that transforms data flow.
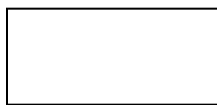
Source or Destination of data

Data flow
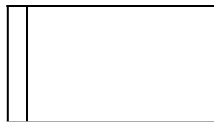
Data Store

CONSTRUCTING A DFD:

Several rules of thumb are used in drawing DFD'S:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
3. When a process is exploded into lower level details, they are numbered.
4. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each work capitalized

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out.

Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

SAILENT FEATURES OF DFD'S

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

TYPES OF DATA FLOW DIAGRAMS

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

CURRENT PHYSICAL:

In Current Physical DFD proecess label include the name of people or their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process the data. Similarly data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

CURRENT LOGICAL:

The physical aspects at the system are removed as mush as possible so that the current system is reduced to its essence to the data and the processors that transform them regardless of actual physical form.

NEW LOGICAL:

This is exactly like a current logical model if the user were completely happy with he user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

NEW PHYSICAL:

The new physical represents only the physical implementation of the new system.

RULES GOVERNING THE DFD'S

PROCESS

1) No process can have only outputs.
2) No process can have only inputs. If an object has only inputs than it must be a sink.
3) A process has a verb phrase label.

DATA STORE

1) Data cannot move directly from one data store to another data store, a process must move data.
2) Data cannot move directly from an outside source to a data store, a process, which receives, must move data from the source and place the data into data store
3) A data store has a noun phrase label.

SOURCE OR SINK

The origin and /or destination of data.

1) Data cannot move direly from a source to sink it must be moved by a process
2) A source and /or sink has a noun phrase land

DATA FLOW

1) A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.

2) A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.

3) A data flow cannot go directly back to the same process it leads. There must be atleast one other process that handles the data flow produce some other data flow returns the original data into the beginning process.

4) A Data flow to a data store means update (delete or change).

5) A data Flow from a data store means retrieve or use.

5.2. UML Diagrams

Use Case Diagram:

- The unified modeling language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

- A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

- User Model View

    i. This view represents the system from the users perspective.

    ii. The analysis representation describes a usage scenario from the end-users perspective.

Structural model view

- In this model the data and functionality are arrived from inside the system.

- This model view models the static structures.

Behavioral model view

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

Implementation Model View

In this the structural and behavioral as parts of the system are represented as they are to be   built.

Environmental Model View

In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

UML is specifically constructed through two different domains they are

- UML Analysis modeling, which focuses on the user model and structural model views of the system.

- UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

Use case Diagrams represent the functionality of the system from a user's point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.

Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer …etc., or another system like central database.

Use case Model
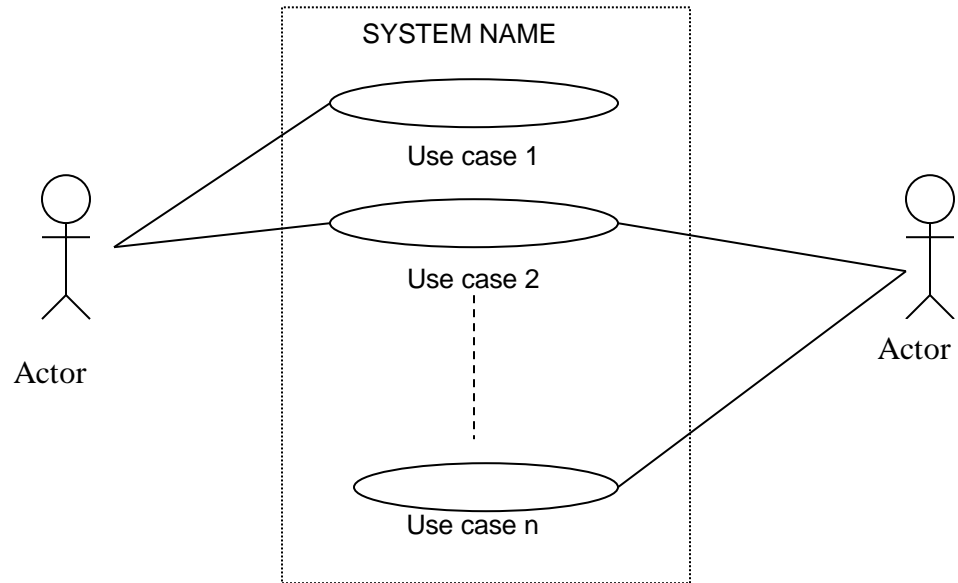
Use Cases of Blood Donation Agent Interface

Use case   For Admin Module

Blood Donation Agent Interface

Admin

Home Page

Login Page

Donar Home Page

Donor Account Detail

Donation Details

Search

Help

Admin Page
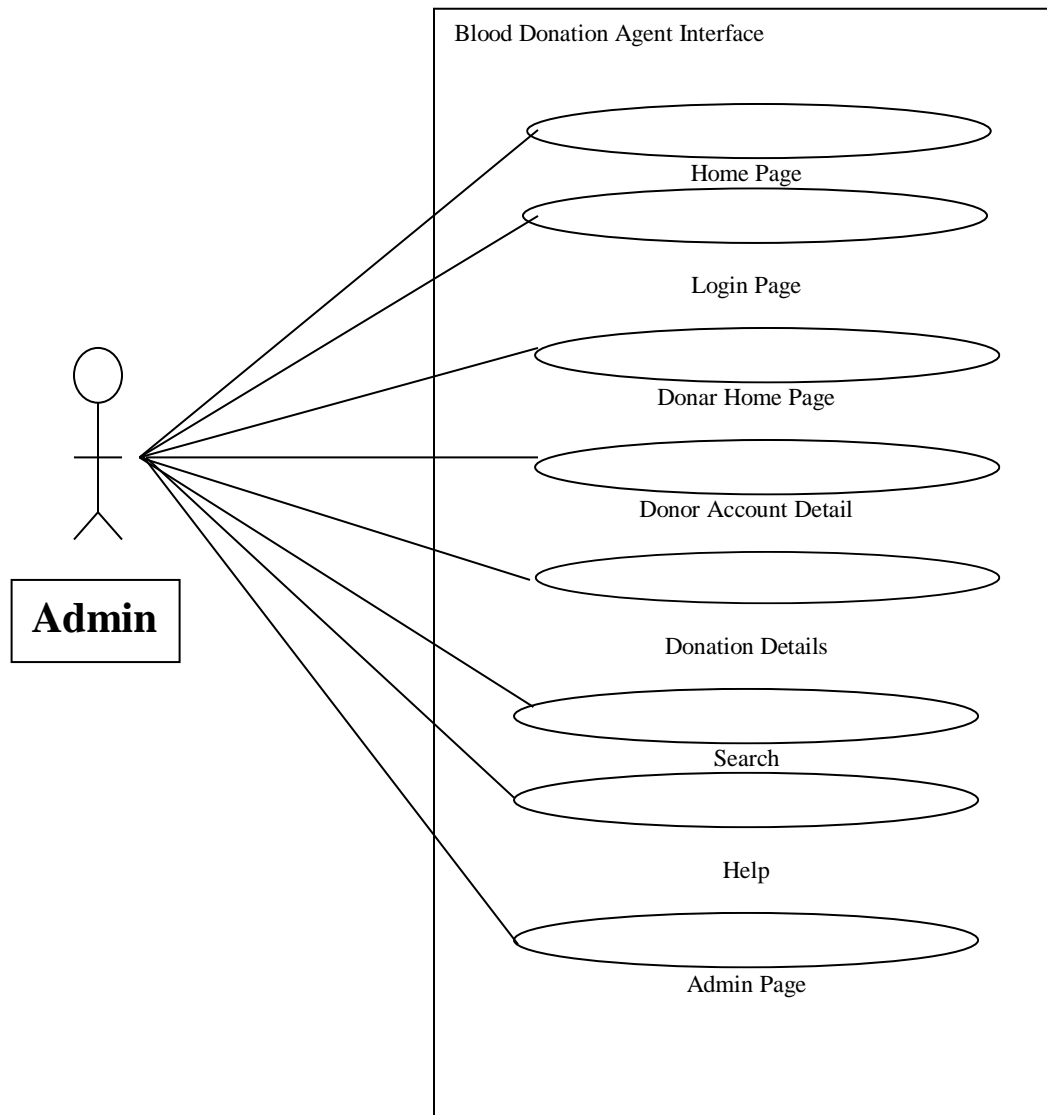
Use case   For Donor Module

Blood Donation Agent Interface

Home Page

Login Page

Donor Home Page

Donor Account Detail

Search Page

Help Page

Donor
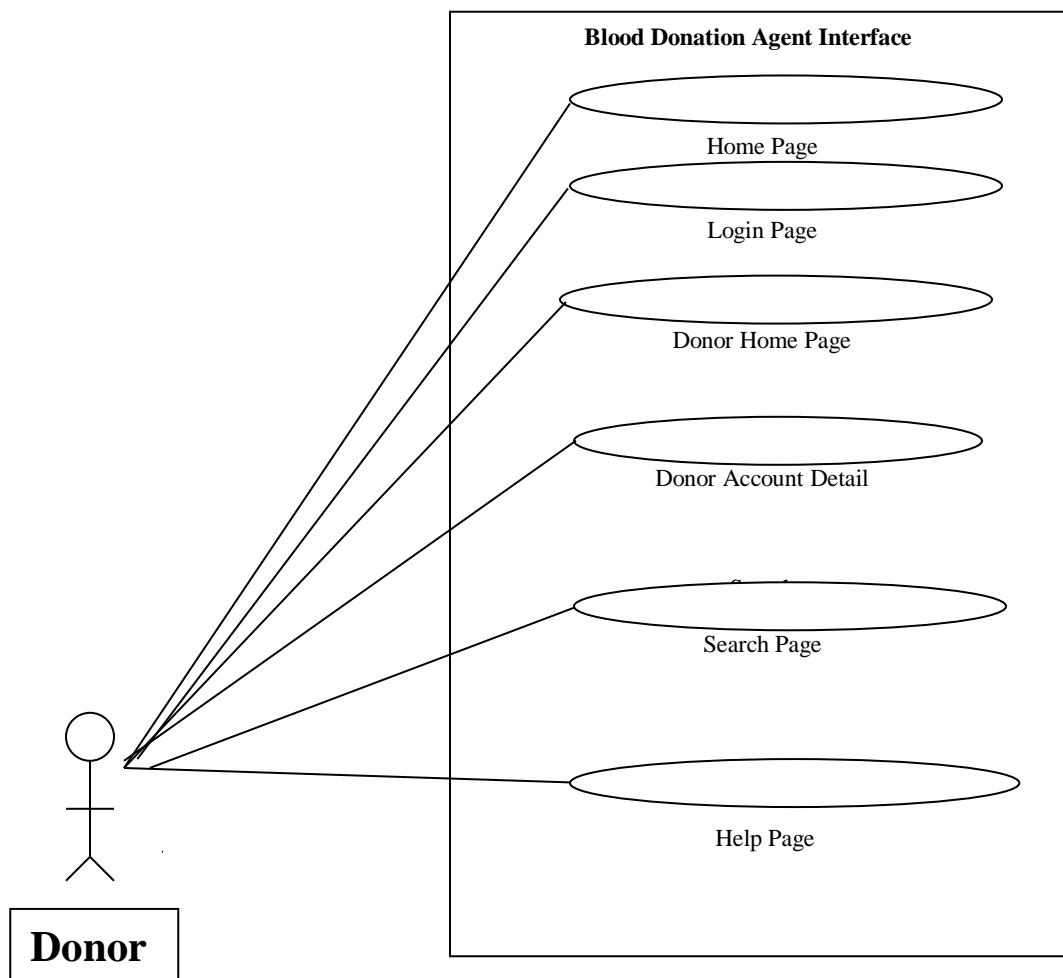
5.3. Database Design

Entities with Attributes:

1.Administration:

User Account:

- AccountID
- Username
- Password

BDA State:

- State Name
- City

City:

- City ID
- City Name

BloodGroup:

- BloodGroupID
- BloodGroup
- Description

PersonalDetails:

- Username
- Name
- Email
- DOB
- Gender
- BloodType
- MobileNo

5.4.ER Diagrams

ER diagram for User Account and Users
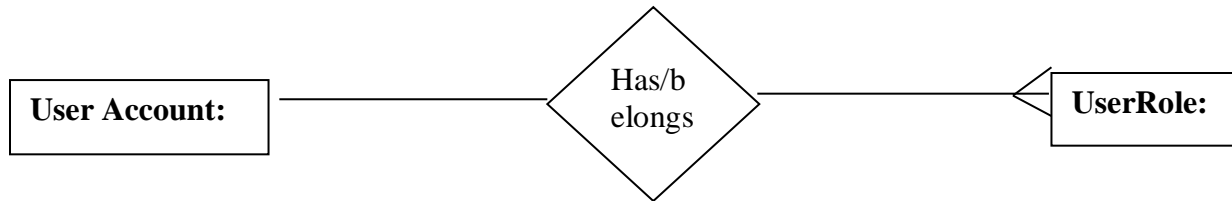
UserAccount:
Username
Password
UserDetails
Updatedetails

UserRole:
RoleID
RoleName

ER diagram for Users Account and Personal Details

Has/belongs

User Account:

Personal details

UserAccount:
name
Username
Password
UpdateDetails

Personal Details:
Name
Email
DOB
Gender
BloodGroup
District
State
MobileNo
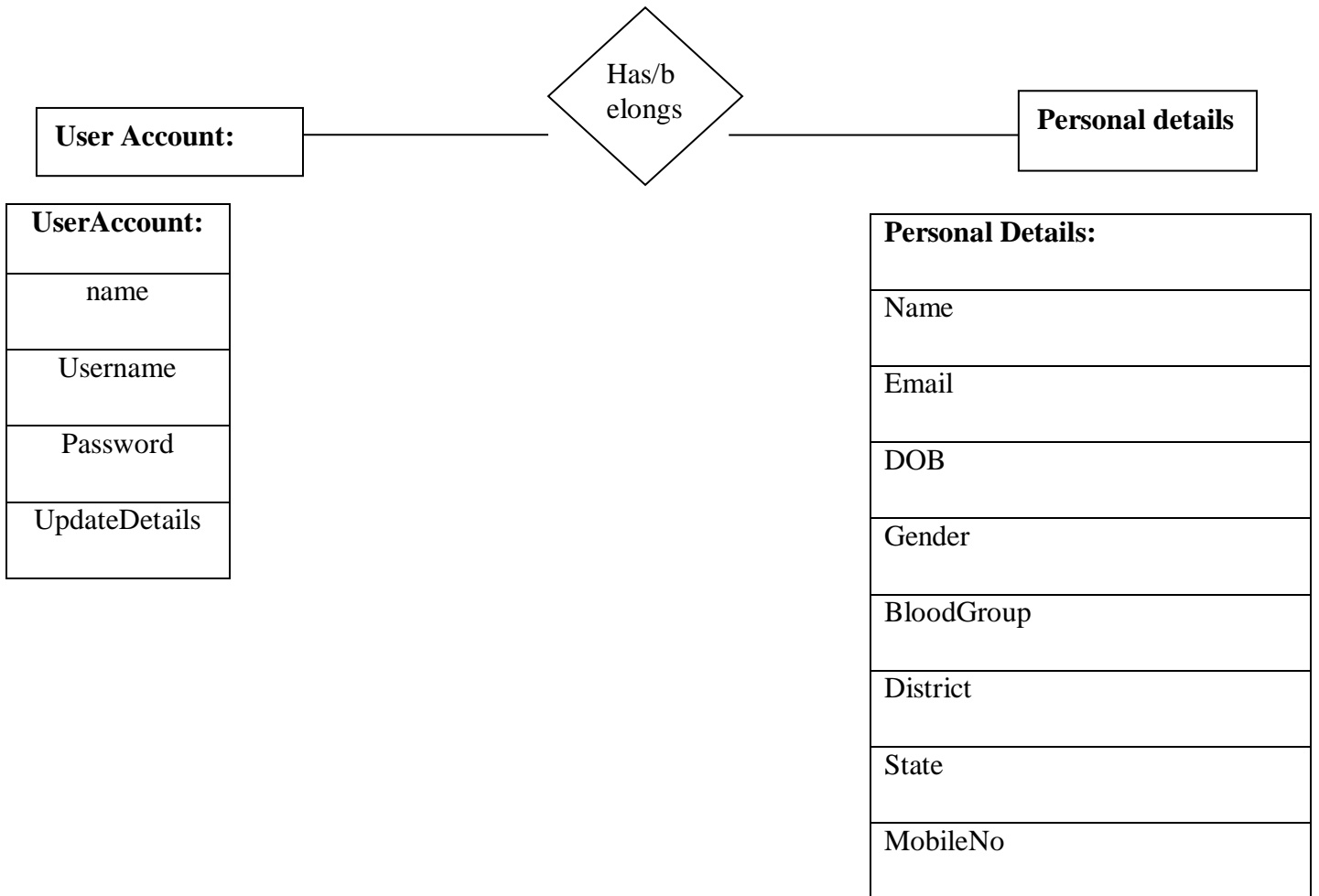
ER diagram for Blood group   Personal details

| Blood group: | Has/b elong | Personal details |
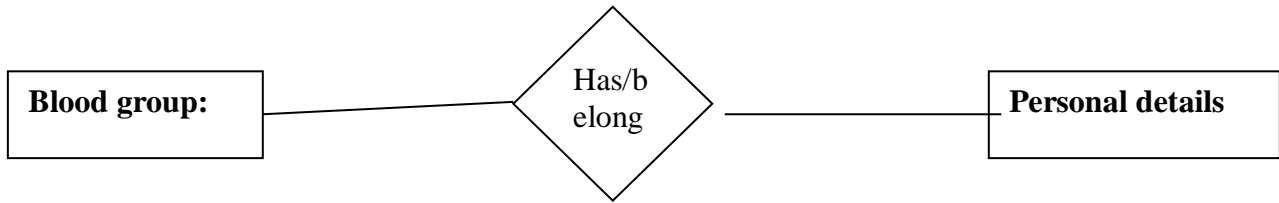
BloodGroup:
BloodGroup
Active

Personal Details:
Name
Email
DOB
Gender
BloodGroup
District
State
MobileNo

5.5. Database Tables

1.Entities

- ➢ DetailTable

2.Entities with Attributes

1. DetailTable
   - Name
   - Username
   - Password
   - Date_of_birth
   - gender
   - blood group
   - mobile_no
   - email
   - state
   - district

2.Data Dictionary

DetailTable

Sno	Column name	Data type	Constraint
1	Name	Varchar(50)	Not null
2	Username	varchar(50)	Primary key
3	Password	varchar(50)	Not null
4	dateOfBirth	date	Not null
5	Gender	varchar(50)	Not null
6	Bloodroup	varchar(50)	Not null
7	MobileNO	Bigint	Not null
8	EmailId	varchar(50)	Not Null
9	State	varchar(50)	Not Null
10	District	varchar(50)	Not Null

Software Development Environment

6.1. Introduction To .Net Framework

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet. The .NET Framework is designed to fulfill the following objectives:

- To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely.
- To provide a code-execution environment that minimizes software deployment and versioning conflicts.
- To provide a code-execution environment that guarantees safe execution of code, including code created by an unknown or semi-trusted third party.
- To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.
- To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.
- To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code.

The .NET Framework has two main components: the common language runtime and the .NET Framework class library. The common language runtime is the foundation of the .NET Framework. You can think of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, and Remoting, while also enforcing strict type safety and other forms of code accuracy that ensure security and robustness. In fact, the concept of code management is a fundamental principle of the runtime. Code that targets the runtime is known as managed code, while code that does not target the runtime is known as unmanaged code.

FEATURES OF THE COMMON LANGUAGE RUNTIME:

The common language runtime manages memory, thread execution, code execution, code safety verification, compilation, and other system services. These features are intrinsic to the managed code that runs on the common language runtime.

With regards to security, managed components are awarded varying degrees of trust, depending on a number of factors that include their origin (such as the Internet, enterprise network, or local computer). This means that a managed component might or might not be able to perform file-access operations, registry-access operations, or other sensitive functions, even if it is being used in the same active application.

The runtime enforces code access security. For example, users can trust that an executable embedded in a Web page can play an animation on screen or sing a song, but cannot access their personal data, file system, or network. The security features of the runtime thus enable legitimate Internet-deployed software to be exceptionally featuring rich.

The runtime also enforces code robustness by implementing a strict type- and code-verification infrastructure called the common type system (CTS). The CTS ensures that all managed code is self-describing. The various Microsoft and third-party language compilers

Generate managed code that conforms to the CTS. This means that managed code can consume other managed types and instances, while strictly enforcing type fidelity and type safety.

In addition, the managed environment of the runtime eliminates many common software issues. For example, the runtime automatically handles object layout and manages references to objects, releasing them when they are no longer being used. This automatic memory management resolves the two most common application errors, memory leaks and invalid memory references.

The runtime also accelerates developer productivity. For example, programmers can write applications in their development language of choice, yet take full advantage of the runtime, the class library, and components written in other languages by other developers. Any compiler vendor who chooses to target the runtime can do so. Language compilers that target the .NET Framework make the features of the .NET Framework available to existing code written in that language, greatly easing the migration process for existing applications.

.NET FRAMEWORK CLASS LIBRARY

The .NET Framework class library is a collection of reusable types that tightly integrate with the common language runtime. The class library is object oriented, providing types from which your own managed code can derive functionality. This not only makes the .NET Framework types easy to use, but also reduces the time associated with learning new features of the .NET Framework. In addition, third-party components can integrate seamlessly with classes in the .NET Framework.

For example, the .NET Framework collection classes implement a set of interfaces that you can use to develop your own collection classes. Your collection classes will blend seamlessly with the classes in the .NET Framework.

As you would expect from an object-oriented class library, the .NET Framework types enable you to accomplish a range of common programming tasks, including tasks such as string management, data collection, database connectivity, and file access. In addition to these common tasks, the class library includes types that support a variety of specialized development scenarios. For example, you can use the .NET Framework to develop the following types of applications and services:

- Console applications.
- Scripted or hosted applications.
- Windows GUI applications (Windows Forms).
- ASP.NET applications.
- XML Web services.
- Windows services.

For example, the Windows Forms classes are a comprehensive set of reusable types that vastly simplify Windows GUI development. If you write an ASP.NET Web Form application, you can use the Web Forms classes.

CLIENT APPLICATION DEVELOPMENT

Client applications are the closest to a traditional style of application in Windows-based programming. These are the types of applications that display windows or forms on the desktop, enabling a user to perform a task. Client applications include applications such as word processors and spreadsheets, as well as custom business applications such as data-entry tools, reporting tools, and so on. Client applications usually employ windows, menus, buttons, and other GUI elements, and they likely access local resources such as the file system and peripherals such as printers.

Another kind of client application is the traditional ActiveX control (now replaced by the managed Windows Forms control) deployed over the Internet as a Web page. This application is much like other client applications: it is executed natively, has access to local resources, and includes graphical elements.

In the past, developers created such applications using C/C++ in conjunction with the Microsoft Foundation Classes (MFC) or with a rapid application development (RAD) environment such as Microsoft® Visual Basic®. The .NET Framework incorporates aspects of these existing products into a single, consistent development environment that drastically simplifies the development of client applications.

The Windows Forms classes contained in the .NET Framework are designed to be used for GUI development. You can easily create command windows, buttons, menus, toolbars, and other screen elements with the flexibility necessary to accommodate shifting business needs.

For example, the .NET Framework provides simple properties to adjust visual attributes associated with forms. In some cases the underlying operating system does not support changing these attributes directly, and in these cases the .NET Framework automatically recreates the forms. This is one of many ways in which the .NET Framework integrates the developer interface, making coding simpler and more consistent.

6.2. ASP.NET

Server Application Development

Server-side applications in the managed world are implemented through runtime hosts. Unmanaged applications host the common language runtime, which allows your custom managed code to control the behavior of the server. This model provides you with all the features of the common language runtime and class library while gaining the performance and scalability of the host server.

The following illustration shows a basic network schema with managed code running in different server environments. Servers such as IIS and SQL Server can perform standard operations while your application logic executes through the managed code.

Server-Side Managed Code:

ASP.NET is the hosting environment that enables developers to use the .NET Framework to target Web-based applications. However, ASP.NET is more than just a runtime host; it is a complete architecture for developing Web sites and Internet-distributed objects using managed code. Both Web Forms and XML Web services use IIS and ASP.NET as the publishing mechanism for applications, and both have a collection of supporting classes in the .NET Framework.

If you develop and publish your own XML Web service, the .NET Framework provides a set of classes that conform to all the underlying communication standards, such as SOAP, WSDL, and XML. Using those classes enables you to focus on the logic of your service, without concerning yourself with the communications infrastructure required by distributed software development.

Finally, like Web Forms pages in the managed environment, your XML Web service will run with the speed of native machine language using the scalable communication of IIS.

6.3. Active Server Pages.NET

ASP.NET is a programming framework built on the common language runtime that can be used on a server to build powerful Web applications. ASP.NET offers several important advantages over previous Web development models:

- **Enhanced Performance.** ASP.NET is compiled common language runtime code running on the server. Unlike its interpreted predecessors, ASP.NET can take advantage of early binding, just-in-time compilation, native optimization, and caching services right out of the box. This amounts to dramatically better performance before you ever write a line of code.

- **World-Class Tool Support.** The ASP.NET framework is complemented by a rich toolbox and designer in the Visual Studio integrated development environment. WYSIWYG editing, drag-and-drop server controls, and automatic deployment are just a few of the features this powerful tool provides.

- **Power and Flexibility.** Because ASP.NET is based on the common language runtime, the power and flexibility of that entire platform is available to Web application developers. The .NET Framework class library, Messaging, and Data Access solutions are all seamlessly accessible from the Web. ASP.NET is also language-independent, so you can choose the language that best applies to your application or partition your application across many languages. Further, common language runtime interoperability guarantees that your existing investment in COM-based development is preserved when migrating to ASP.NET.

- **Simplicity.** ASP.NET makes it easy to perform common tasks, from simple form submission and client authentication to deployment and site configuration. For example, the ASP.NET page framework allows you to build user interfaces that cleanly separate application logic from presentation code and to handle events in a simple, Visual Basic - like forms processing model. Additionally, the common language runtime simplifies development, with managed code services such as automatic reference counting and garbage collection.

- **Manageability.** ASP.NET employs a text-based, hierarchical configuration system, which simplifies applying settings to your server environment and Web applications. Because configuration information is stored as plain text, new settings may be applied without the aid of local administration tools. This "zero local administration" philosophy extends to deploying ASP.NET

- **Scalability and Availability.** ASP.NET has been designed with scalability in mind, with features specifically tailored to improve performance in clustered and multiprocessor environments. Further, processes are closely monitored and managed by the ASP.NET runtime, so that if one misbehaves (leaks, deadlocks), a new process can be created in its place, which helps keep your application constantly available to handle requests.

- **Customizability and Extensibility.** ASP.NET delivers a well-factored architecture that allows developers to "plug-in" their code at the appropriate level. In fact, it is possible to extend or replace any subcomponent of the ASP.NET runtime with your own custom-written component. Implementing custom authentication or state services has never been easier.

- **Security.** With built in Windows authentication and per-application configuration, you can be assured that your applications are secure.

6.4. LANGUAGE SUPPORT

The Microsoft .NET Platform currently offers built-in support for three languages: C#, Visual Basic, and JScript.

WHAT IS ASP.NET WEB FORMS?

The ASP.NET Web Forms page framework is a scalable common language runtime programming model that can be used on the server to dynamically generate Web pages.

Intended as a logical evolution of ASP (ASP.NET provides syntax compatibility with existing pages), the ASP.NET Web Forms framework has been specifically designed to address a number of key deficiencies in the previous model. In particular, it provides:

- The ability to create and use reusable UI controls that can encapsulate common functionality and thus reduce the amount of code that a page developer has to write.

- The ability for developers to cleanly structure their page logic in an orderly fashion (not "spaghetti code").

- The ability for development tools to provide strong WYSIWYG design support for pages (existing ASP code is opaque to tools).

6.5. CODE-BEHIND WEB FORMS

ASP.NET supports two methods of authoring dynamic pages. The first is the method shown in the preceding samples, where the page code is physically declared within the originating .aspx file. An alternative approach--known as the code-behind method--enables the page code to be more cleanly separated from the HTML content into an entirely separate file.

6.6. C#.NETand ADO.NET OVERVIEW

ADO.NET is an evolution of the ADO data access model that directly addresses user requirements for developing scalable applications. It was designed specifically for the web with scalability, statelessness, and XML in mind.

ADO.NET uses some ADO objects, such as the **Connection** and **Command** objects, and also introduces new objects. Key new ADO.NET objects include the **DataSet**, **DataReader**, and **DataAdapter**.

The important distinction between this evolved stage of ADO.NET and previous data architectures is that there exists an object -- the **DataSet** -- that is separate and distinct from any data stores. Because of that, the **DataSet** functions as a standalone entity. You can think of the DataSet as an always disconnected recordset that knows nothing about the source or destination of the data it contains. Inside a **DataSet**, much like in a database, there are tables, columns, relationships, constraints, views, and so forth.

A **DataAdapter** is the object that connects to the database to fill the **DataSet**. Then, it connects back to the database to update the data there, based on operations performed while the **DataSet** held the data. In the past, data processing has been primarily connection-based. Now, in an effort to make multi-tiered apps more efficient, data processing is turning to a message-based approach that revolves around chunks of information. At the center of this approach is the **DataAdapter**, which provides a bridge to retrieve and save data between a **DataSet** and its source data store. It accomplishes this by means of requests to the appropriate SQL commands made against the data store.

- **Connections**. For connection to and managing transactions against a database.
- **Commands**. For issuing SQL commands against a database.
- **DataReaders**. For reading a forward-only stream of data records from a SQL Server data source.
- **DataSets**. For storing, Remoting and programming against flat data, XML data and relational data.

- **Data Adapters**. For pushing data into a **DataSet**, and reconciling data against a database.

Connections:

Connections are used to 'talk to' databases, and are represented by provider-specific classes such as **SqlConnection**. Commands travel over connections and resultsets are returned in the form of streams which can be read by a **DataReader** object, or pushed into a **DataSet** object.

Commands:

Commands contain the information that is submitted to a database, and are represented by provider-specific classes such as **SqlCommand**. A command can be a stored procedure call, an UPDATE statement, or a statement that returns results. You can also use input and output parameters, and return values as part of your command syntax. The example below shows how to issue an INSERT statement against the **Northwind** database.

DataReaders:

The **DataReader** object is somewhat synonymous with a read-only/forward-only cursor over data. The **DataReader** API supports flat as well as hierarchical data. A **DataReader** object is returned after executing a command against a database. The format of the returned **DataReader** object is different from a recordset. For example, you might use the **DataReader** to show the results of a search list in a web page.

6.7. SQL SERVER

A database management, or DBMS, gives the user access to their data and helps them transform the data into information. Such database management systems include dBase, paradox, IMS, SQL Server and SQL Server. These systems allow users to create, update and extract information from their database.

A database is a structured collection of data. Data refers to the characteristics of people, things and events. SQL Server stores each data item in its own fields. In SQL Server, the fields relating to a particular person, thing or event are bundled together to form a single complete unit of data, called a record (it can also be referred to as raw or an occurrence). Each record is made up of a number of fields. No two fields in a record can have the same field name.

During an SQL Server Database design project, the analysis of your business needs identifies all the fields or attributes of interest. If your business needs change over time, you define any additional fields or change the definition of existing fields.

SQL Server Tables

SQL Server stores records relating to each other in a table. Different tables are created for the various groups of information. Related tables are grouped together to form a database.

Primary Key

Every table in SQL Server has a field or a combination of fields that uniquely identifies each record in the table. The Unique identifier is called the Primary Key, or simply the Key. The primary key provides the means to distinguish one record from all other in a table. It allows the user and the database system to identify, locate and refer to one particular record in the database.

Relational Database

Sometimes all the information of interest to a business operation can be stored in one table. SQL Server makes it very easy to link the data in multiple tables. Matching an employee to the department in which they work is one example. This is what makes SQL Server a relational database management system, or RDBMS. It stores data in two or more tables and enables you to define relationships between the table and enables you to define relationships between the tables.

Foreign Key

When a field is one table matches the primary key of another field is referred to as a foreign key. A foreign key is a field or a group of fields in one table whose values match those of the primary key of another table.

Referential Integrity

Not only does SQL Server allow you to link multiple tables, it also maintains consistency between them. Ensuring that the data among related tables is correctly matched is referred to as maintaining referential integrity.

CODING

7.1. WEBCONFIG FILE:(Design Code)

Used to set the connections of each page.

```xml
<?xml version="1.0"?>

<!--
  For more information on how to configure your ASP.NET application, please visit
  http://go.microsoft.com/fwlink/?LinkId=169433
  -->

<configuration>
    <appSettings>
    <add key="ValidationSettings:UnobtrusiveValidationMode" value="None"></add>
  </appSettings>
    <connectionStrings>
        <add name="ConnectionString" connectionString="Data
Source=(LocalDB)\v11.0;AttachDbFilename=|DataDirectory|\register.mdf;Integrated
Security=True"
            providerName="System.Data.SqlClient" />
    </connectionStrings>
    <system.web>
      <compilation debug="true" targetFramework="4.5" />
      <httpRuntime targetFramework="4.5" />
    </system.web>
  <system.webServer>
    <defaultDocument enabled="true">
      <files>
        <clear />
        <add value="home.aspx"/>
      </files>
    </defaultDocument>
  </system.webServer>

</configuration>
```

7.2. donarAccount Form

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
using System.Configuration;

namespace BloodDonationManagementSystem
{
    public partial class WebForm3 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            string query = "select username ,pasword from registertable where username='"
+ TextBox1.Text + "' and pasword='" + TextBox2.Text + "'";
            SqlConnection con = new SqlConnection();
            con.ConnectionString = constring.cstr;
            con.Open();
            SqlCommand cmd = new SqlCommand(query, con);
            SqlDataReader dr = cmd.ExecuteReader();
            if (dr.Read())
            {
                Session["name"] = TextBox1.Text;
                Response.Redirect("updateRegister.aspx");

            }
            else
            {
                Response.Write("<script>alert('Please enter valid Username and
Password')</script>");
            }
            con.Close();
        }

    }
}
```

7.3. Registeration Form

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
using System.Configuration;

namespace BloodDonationManagementSystem
{
    public partial class WebForm5 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }
        protected void Button1_Click(object sender, EventArgs e)
        {
            String gender=" ";
            if (RadioButton1.Checked)
                gender = "male";
            else
                gender = "female";
            string query = "select count(*) from registertable where username='" +
TextBox2.Text + "'";
            SqlConnection con = new SqlConnection();
            con.ConnectionString = constring.cstr;
            con.Open();
            SqlCommand cmd = new SqlCommand(query, con);
            int OBJ = Convert.ToInt32(cmd.ExecuteScalar());
            if (OBJ == 1)
            {
                Response.Write("<script>alert('username already exists')</script>");
            }
            else
            {
                string query1 = "insert into registertable values ('" + TextBox1.Text +
"' ,'" + TextBox2.Text + "' ,'" + TextBox3.Text + "','" + TextBox4.Text + "','" + gender
+ "','" + DropDownList1.Text + "','" + TextBox5.Text + "','" + TextBox6.Text + "','" +
DropDownList2.Text + "','" + DropDownList3.Text + "')";
                SqlConnection con1 = new SqlConnection();
                con1.ConnectionString = constring.cstr;
                con1.Open();
                SqlCommand cmd1 = new SqlCommand(query1, con1);
                cmd1.ExecuteNonQuery();
                Label11.Visible = true;
                Label11.Text = "you have registered succesfully";
            //   Response.Write("<script>alert('you have registered
succesfully')</script>");
```

```csharp
            // Response.Redirect("donarAccount.aspx");
                con1.Close();
            }
            con.Close();
        }


    protected void CustomValidator1_ServerValidate(object source,
ServerValidateEventArgs e)
        {
            if ((e.Value.Length >= 8) && (e.Value.Length <=15))
                e.IsValid = true;
            else
                e.IsValid = false;
        }
    }
}
```

7.4. SEARCH FORM

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
using System.Configuration;
using System.Data;

namespace BloodDonationManagementSystem
{
    public partial class WebForm4 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            string query = "select
name,date_of_birth,gender,blood_group,mobile_no,email_id,state,district from
registertable where blood_group='"+DropDownList1.Text+"' AND
state='"+DropDownList2.Text+"' AND district='"+DropDownList3.Text+"'";
            SqlConnection con = new SqlConnection();
            con.ConnectionString = constring.cstr;
            con.Open();
            SqlCommand cmd = new SqlCommand(query, con);
            cmd.ExecuteNonQuery();
            SqlDataAdapter da = new SqlDataAdapter();
            da.SelectCommand = cmd;
            DataSet ds = new DataSet();
            da.Fill(ds);
            GridView1.DataSource = ds;
            GridView1.DataBind();
            con.Close();
        }

        protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
        {

        }
    }
}
```
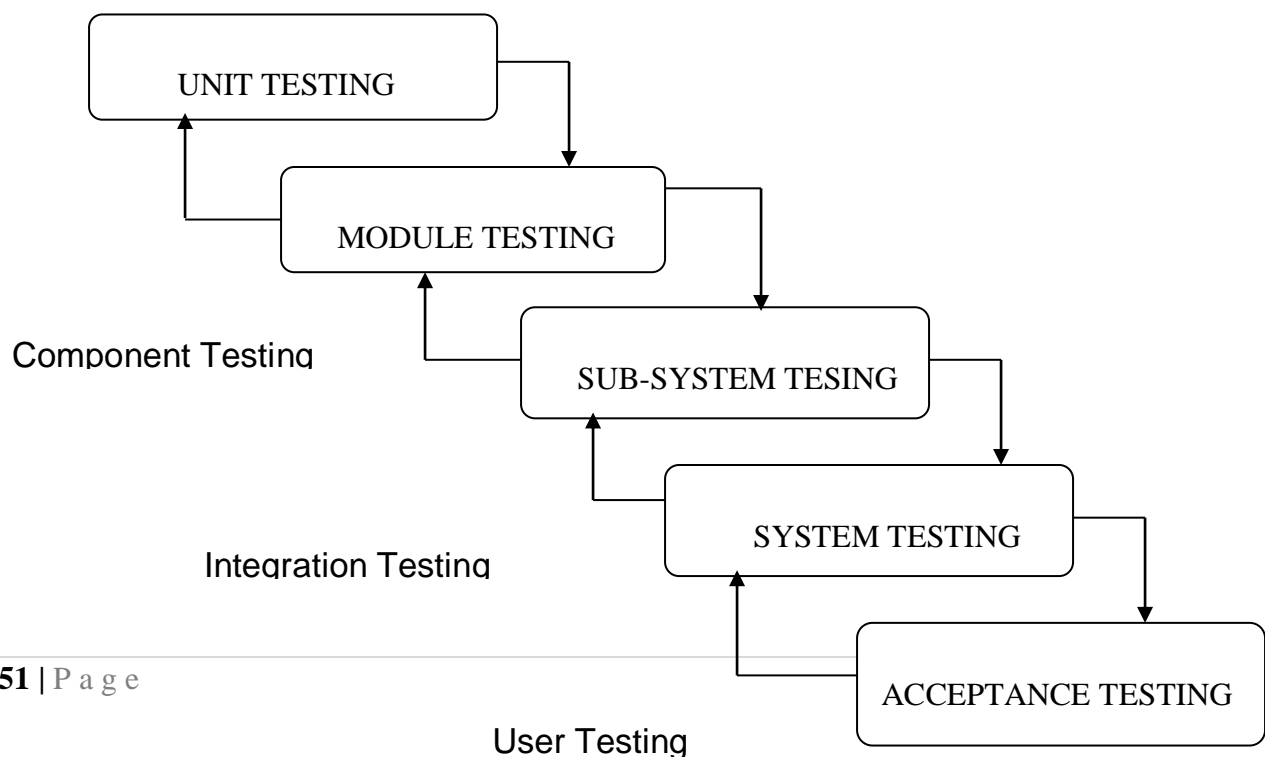
Testing

8.1. INTRODUCTION

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

8.2. STRATEGIC APPROACH TO SOFTWARE TESTING

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

8.3. Unit Testing

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel.

1. WHITE BOX TESTING

This type of testing ensures that

- All independent paths have been exercised at least once
- All logical decisions have been exercised on their true and false sides
- All loops are executed at their boundaries and within their operational bounds
- All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form .we have created independently to verify that Data flow is correct, All conditions are exercised to check their validity, All loops are executed on their boundaries.

2. BASIC PATH TESTING

Established technique of flow graph with Cyclomatic complexity was used to derive test cases for all the functions. The main steps in deriving test cases were:

Use the design of the code and draw correspondent flow graph.

Determine the Cyclomatic complexity of resultant flow graph, using formula:

$V(G)=E-N+2$ or

$V(G)=P+1$ or

$V(G)=$Number Of Regions

Where $V(G)$ is Cyclomatic complexity,

E is the number of edges,

N is the number of flow graph nodes,

P is the number of predicate nodes.

Determine the basis of set of linearly independent paths.

3. CONDITIONAL TESTING

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generate on particular condition is traced to uncover any possible errors.

4. DATA FLOW TESTING

This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable were declared. The *definition-use chain* method was used in this type of testing. These were particularly useful in nested statements.
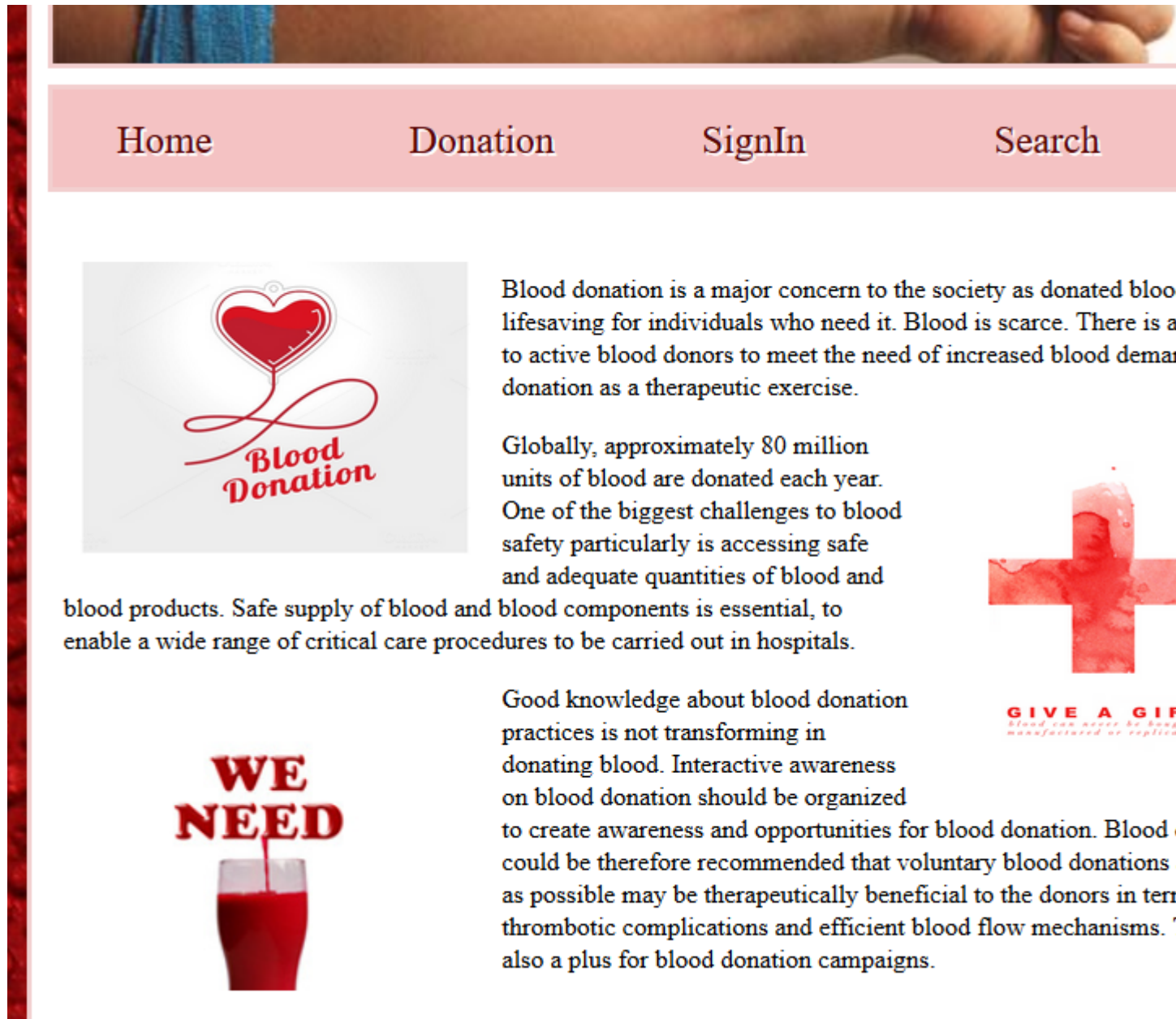
5. LOOP TESTING

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

- All the loops were tested at their limits, just above them and just below them.
- All the loops were skipped at least once.
- For nested loops test the inner most loop first and then work outwards.
- For concatenated loops the values of dependent loops were set with the help of connected loop.
- Unstructured loops were resolved into nested loops or concatenated loops and tested as above.

Each unit has been separately tested by the development team itself and all the input have been validated.

Output Screens

9.1. Home Page

Home   Donation   SignIn   Search

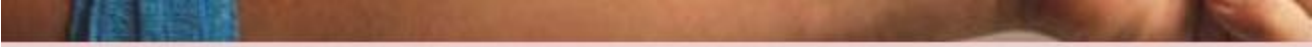Blood donation is a major concern to the society as donated bloo[...] lifesaving for individuals who need it. Blood is scarce. There is a[...] to active blood donors to meet the need of increased blood dema[...] donation as a therapeutic exercise.

Globally, approximately 80 million units of blood are donated each year. One of the biggest challenges to blood safety particularly is accessing safe and adequate quantities of blood and blood products. Safe supply of blood and blood components is essential, to enable a wide range of critical care procedures to be carried out in hospitals.

Good knowledge about blood donation practices is not transforming in donating blood. Interactive awareness on blood donation should be organized to create awareness and opportunities for blood donation. Blood [...] could be therefore recommended that voluntary blood donations [...] as possible may be therapeutically beneficial to the donors in ter[...] thrombotic complications and efficient blood flow mechanisms. [...] also a plus for blood donation campaigns.

9.2. Admin Login

UserName

Password

LogIn

9.3. Admin Home

Home   Donation   SignIn   Search

	name	username	pasword	date_of_birth	gender	blood_group	mobile_no	email_id	state
Edit Delete	amit	amit	asdf	07-May-05 12:00:00 AM	male	A+	8800894352	sfdk@gmail.com	Uttar Pradesh
Edit Delete	amit	amitguptaf	gerhzeherah	07-May-05 12:00:00 AM	female	B+	1212121212	amitgg@gmail.com	Uttar Pradesh
Edit Delete	asdffgg	dys	1443434343	01-Dec-93 12:00:00 AM	male	A+	9999999999	nip@gmail.com	Uttar Pradesh
Edit Delete	rahul	dyss	1211211212	01-Jan-01 12:00:00 AM	male	+	8800232323	RAHUL@gmail.com	Uttar pradesh
Edit Delete	eer	fes	y6465465465	07-May-05 12:00:00 AM	male	A-	7656354675	gfdjhbvhgd@jhghg.com	Uttar Pradesh
Edit Delete	nipurn	nipurncool	nipurn	01-Jan-01 12:00:00 AM	male	A+	8898898989	nip@gmail.com	Uttar Pradesh

9.4. Registration Form For All Users

9.5.Login form

9.6. User Details

NAME	amit
USERNAME	amit
date of birth	07-May-05 12:00:00 AM
gender	male
blood group	A+
mobile no	6565656565
email	sfdk@gmail.com
state	Uttar Pradesh
district	Ghaziabad

logout

9.7 Search For Donor

search for blood donar

blood group   A+ ⌄

state   Uttar Pradesh ⌄

district   Ghaziabad ⌄

submit

name	date_of_birth	gender	blood_group	mobile_no	email_id	state	district
amit	07-May-05 12:00:00 AM	male	A+	6565656565	sfdk@gmail.com	Uttar Pradesh	Ghaziabad
amit	07-May-05 12:00:00 AM	male	A+	1212121212	amitgg@gmail.com	Uttar Pradesh	Ghaziabad
asdffgg	01-Dec-93 12:00:00 AM	male	A+	9999999999	nip@gmail.com	Uttar Pradesh	Ghaziabad

9.8. Update Account Details

Donation   **SignIn**   **Search**

EDIT your DETAILS

UserName	amit
password	
NewPassword	
Mobile no	
Email ID	

Update   LOGOUT

UserDetails

CONCLUSION

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only programming in ASP.NET and VB.NET web based application and no some extent Windows Application and SQL Server, but also about all handling procedure related with **"Blood Bequeath Federal".** It also provides knowledge about the latest technology used in developing web enabled application and client server technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.

10.1. BENEFITS:

The project is identified by the merits of the system offered to the user. The merits of this project are as follows: -

- It's a web-enabled project.

- This project offers user to enter the data through simple and interactive forms. This is very helpful for the client to enter the desired information through so much simplicity.

- The user is mainly more concerned about the validity of the data, whatever he is entering. There are checks on every stages of any new creation, data entry or updation so that the user cannot enter the invalid data, which can create problems at later date.

- Sometimes the user finds in the later stages of using project that he needs to update some of the information that he entered earlier. There are options for him by which he can update the records. Moreover there is restriction for his that he cannot change the primary data field. This keeps the validity of the data to longer extent.

- User is provided the option of monitoring the records he entered earlier. He can see the desired records with the variety of options provided by him.

- From every part of the project the user is provided with the links through framing so that he can go from one option of the project to other as per the requirement. This is bound to be

simple and very friendly as per the user is concerned. That is, we can sat that the project is user friendly which is one of the primary concerns of any good project.

- Data storage and retrieval will become faster and easier to maintain because data is stored in a systematic manner and in a single database.

- Decision making process would be greatly enhanced because of faster processing of information since data collection from information available on computer takes much less time then manual system.

- Allocating of sample results becomes much faster because at a time the user can see the records of last years.

- Easier and faster data transfer through latest technology associated with the computer and communication.

- Through these features it will increase the efficiency, accuracy and transparency,

10.2. LIMITATIONS:

- The size of the database increases day-by-day, increasing the load on the database back up and data maintenance activity.

- Training for simple computer operations is necessary for the users working on the system.

BIBLIOGRAPHY

- **FOR .NET INSTALLATION**

  www.support.mircosoft.com

- **FOR DEPLOYMENT AND PACKING ON SERVER**

  www.developer.com

  www.15seconds.com

- **FOR SQL**

  www.msdn.microsoft.com

- **FOR ASP.NET**

  www.msdn.microsoft.com/net/quickstart/aspplus/default.com

  www.asp.net

  www.fmexpense.com/quickstart/aspplus/default.com

  www.asptoday.com

  www.aspfree.com

  www.4guysfromrolla.com/index.aspx

Reference

Acharya, Kamal. "STUDENT INFORMATION MANAGEMENT SYSTEM." Authorea Preprints (2023).

Acharya, Kamal. "Library Management System." Available at SSRN4807104 (2019).

ACHARYA, KAMAL, et al. "LIBRARY MANAGEMENT SYSTEM." (2019).

Acharya, Kamal. "Online bus reservation system project report." Authorea Preprints (2024).

Acharya, Kamal. "Online bus reservation system project report." (2024).

Acharya, Kamal. "Online Bus Reservation System." SSRN ElectroNIC ASIA Journal (2024): n. pag.

Acharya, Kamal. "Student Information Management System Project." SSRN ElectroNIC ASIA Journal (2024): n. pag.

Acharya, Kamal. "ATTENDANCE MANAGEMENT SYSTEM." International Research Journal of Modernization in Engineering Technology and Science (2023): n. pag.

Acharya, Kamal. "College Information Management System." SSRN ElectroNIC ASIA Journal (2024): n. pag.

Acharya, Kamal, Attendance Management System Project (April 28, 2024). Available at SSRN: https://ssrn.com/abstract=4810251 or http://dx.doi.org/10.2139/ssrn.4810251

Acharya, Kamal, Online Food Order System (May 2, 2024). Available at SSRN: https://ssrn.com/abstract=4814732 or http://dx.doi.org/10.2139/ssrn.4814732

Acharya, Kamal, University management system project. (May 1, 2024). Availableat SSRN: https://ssrn.com/abstract=4814103 or http://dx.doi.org/10.2139/ssrn.4814103

Acharya, Kamal, Online banking management system. (May 1, 2024). Available at SSRN: https://ssrn.com/abstract=4813597 or http://dx.doi.org/10.2139/ssrn.4813597

Acharya, Kamal, Online Job Portal Management System (May 5, 2024). Available at SSRN: https://ssrn.com/abstract=4817534 or http://dx.doi.org/10.2139/ssrn.4817534

Acharya, Kamal, Employee leave management system. (May 7, 2024). Available at SSRN: https://ssrn.com/abstract=4819626 or http://dx.doi.org/10.2139/ssrn.4819626

Acharya, Kamal, Online electricity billing project report. (May 7, 2024). Available at SSRN: https://ssrn.com/abstract=4819630 or http://dx.doi.org/10.2139/ssrn.4819630

Acharya, Kamal, POLICY MANAGEMENT SYSTEM PROJECT REPORT. (December 10, 2023). Available at SSRN: https://ssrn.com/abstract=4831694 or http://dx.doi.org/10.2139/ssrn.4831694

Acharya, Kamal, Online job placement system project report. (January 10, 2023). Available at SSRN: https://ssrn.com/abstract=4831638 or http://dx.doi.org/10.2139/ssrn.4831638

Acharya, Kamal, Software testing for project report. (May 16, 2023). Available at SSRN: https://ssrn.com/abstract=4831028 or http://dx.doi.org/10.2139/ssrn.4831028

Acharya, Kamal, ONLINE CRIME REPORTING SYSTEM PROJECT. (August 10, 2022).
Available at
SSRN: https://ssrn.com/abstract=4831015 or http://dx.doi.org/10.2139/ssrn.4831015
Acharya, Kamal, Burber ordering system project report. (October 10, 2022). Available at
SSRN: https://ssrn.com/abstract=4832704 or http://dx.doi.org/10.2139/ssrn.4832704
Acharya, Kamal, Teachers Record Management System Project Report (December 10,
2023). Available at
SSRN: https://ssrn.com/abstract=4833821 or http://dx.doi.org/10.2139/ssrn.4833821
Acharya, Kamal, Dairy Management System Project Report (December 20, 2020).
Available at
SSRN: https://ssrn.com/abstract=4835231 or http://dx.doi.org/10.2139/ssrn.4835231

Acharya, Kamal, Electrical Shop Management System Project (December 10, 2019).
Available at
SSRN: https://ssrn.com/abstract=4835238 or http://dx.doi.org/10.2139/ssrn.4835238
Acharya, Kamal, Online book store management system project report. (Febuary 10,
2020). Available at
SSRN: https://ssrn.com/abstract=4835277 or http://dx.doi.org/10.2139/ssrn.4835277
Acharya, Kamal, Paint shop management system project report. (January 10, 2019).
Available at
SSRN: https://ssrn.com/abstract=4835441 or http://dx.doi.org/10.2139/ssrn.4835441
Acharya, Kamal, Supermarket billing system project report. (August 10, 2021). Available

at SSRN: https://ssrn.com/abstract=4835474 or http://dx.doi.org/10.2139/ssrn.4835474

Acharya, Kamal, Online texi booking system project report. (March 10, 2022). Available

at SSRN: https://ssrn.com/abstract=4837729 or http://dx.doi.org/10.2139/ssrn.4837729

Acharya, Kamal, Online car servicing system project report. (March 10, 2023). Available

at SSRN: https://ssrn.com/abstract=4837832 or http://dx.doi.org/10.2139/ssrn.4837832

Acharya, Kamal, School management system project report. (July 10, 2021). Available at

SSRN: https://ssrn.com/abstract=4837837 or http://dx.doi.org/10.2139/ssrn.4837837

Acharya, Kamal, Furniture Showroom Management System Project Report (March 21, 2021).

Available at

SSRN: https://ssrn.com/abstract=4839422 or http://dx.doi.org/10.2139/ssrn.4839422

Acharya, Kamal, Online Vehicle Rental System Project Report (March 21, 2019). Available at

SSRN: https://ssrn.com/abstract=4839429 or http://dx.doi.org/10.2139/ssrn.4839429